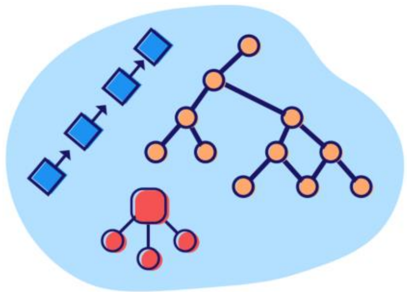
الجامعة العربية الخاصة للعلوم والتكنولوجيا
كلية الهندسة المعلوماتية
قسم تقانة المعلومات

Algorithms & Data Structures (2)

Lecture-1: Graph
(البيان)



Dr. Asmaa Shaar

محتوى المقرر

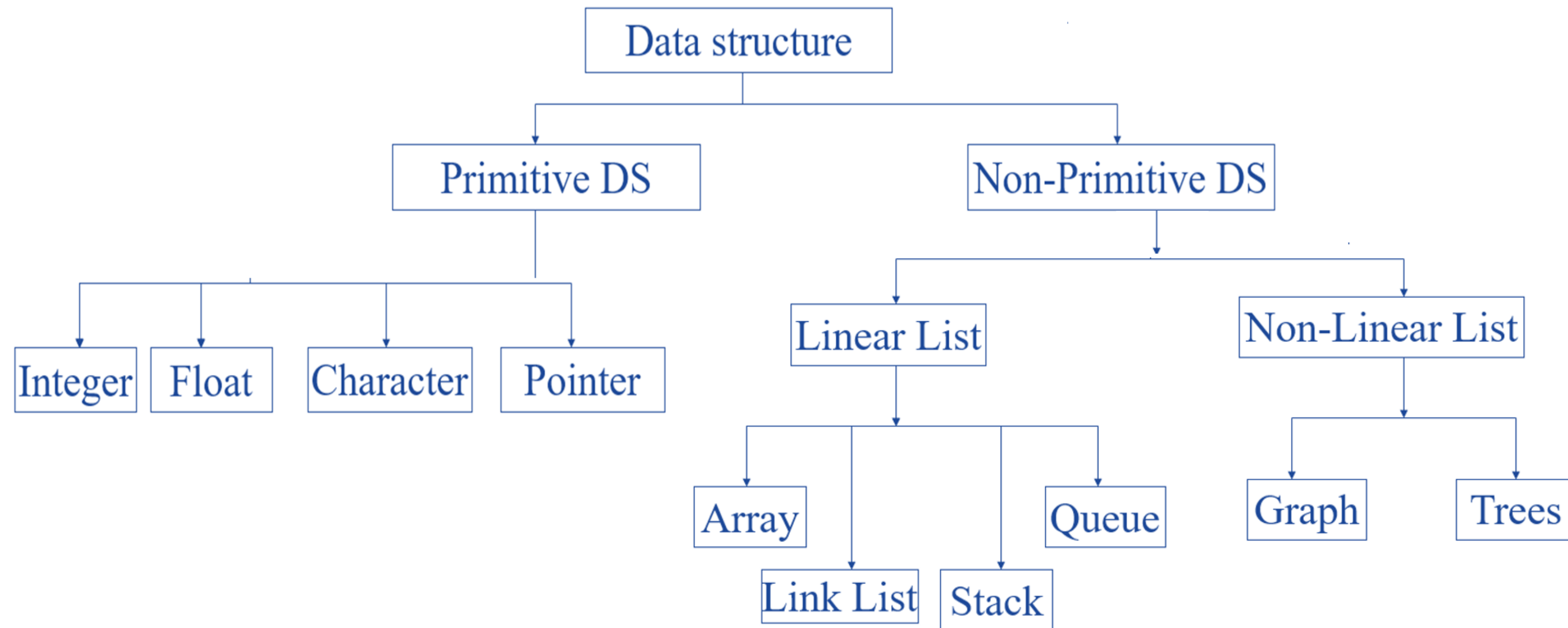
- Non-linear data structures
  - Graph
  - Trees
- Hashing Tables
- Greedy Algorithms
- Backtracking Algorithms
- Dynamic Programming
- Divide and Conquer

فهرس المحاضرة

- مقدمة
- البيان (graph).
  - تعاريف أساسية
    - البيان الكامل (complete graph)
    - البيان الجزئي (sub-graph)
    - البيان المترابط (Connected Graph)
    - درجة رأس (Node Degree)
  - تمثيل البيان:
    - باستخدام مصفوفة التجاور
    - باستخدام قوائم التجاور
  - البيان الموزون

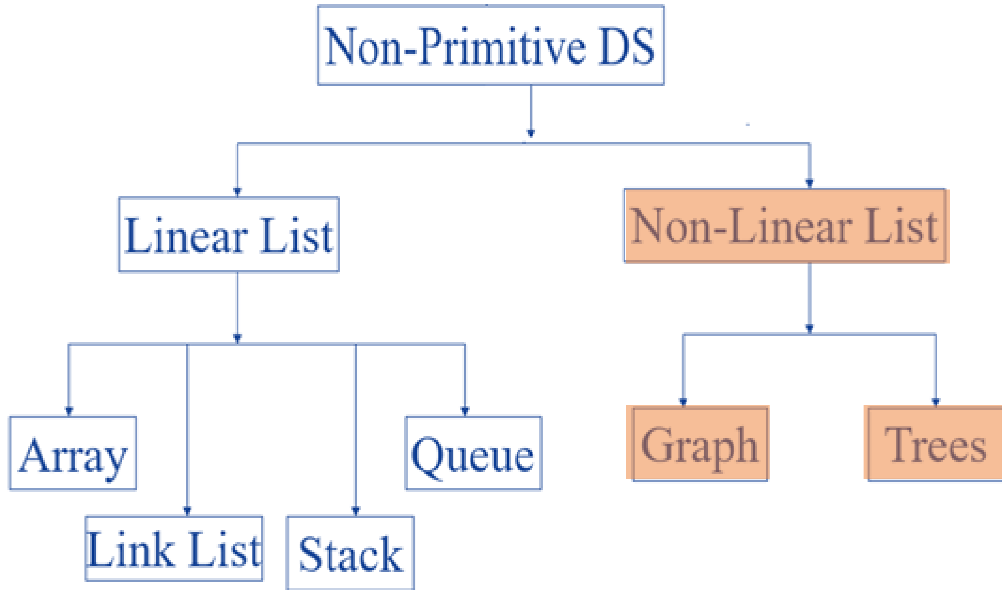
مقدمة

- تنقسم بنى المعطيات إلى فئتين رئيسيتين:



- Primitive DS: هي البنى الأساسية التي تكون مبنية داخلياً (built–in) ضمن أي لغة برمجة.
  - تمت دراسة معظم البنى الأساسية في مقرر البرمجة لذا سوف يكون تركيزنا على (Non-Primitive DS).

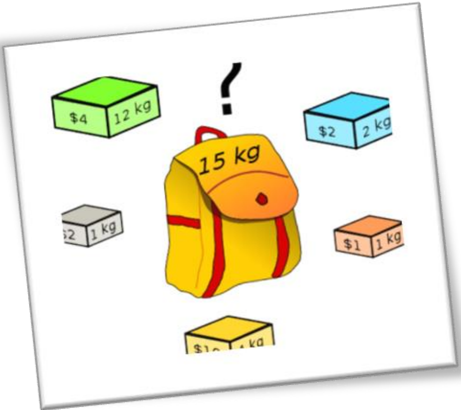
مقدمة

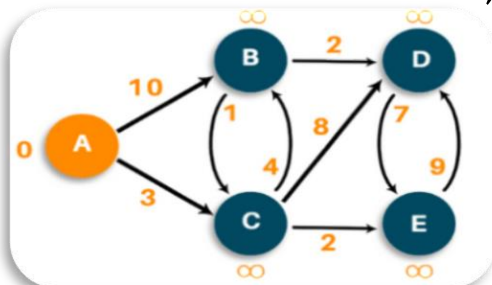


Non−primitive data structures	
Non−linear DS	**linear DS**
يتم ترتيب العناصر بطريقة غير تسلسلية، ويمكن أن يرتبط العنصر بعدة عناصر.	يتم ترتيب عناصر البيانات بشكل تسلسلي، بحيث يتم ربط كل عنصر بالعناصر المجاورة له (العنصر السابق والتالي).
لدينا عدة مستويات.	لدينا مستوى وحيد (single level).
تتطلب عدة جولات للمرور على جميع العناصر.	يمكننا المرور (زيارة) على كل العناصر في جولة واحدة فقط.
صعبة التحقيق بالمقارنة مع بنى المعطيات الخطية	سهلة التحقيق.
يتم استخدام الذاكرة بطريقة فعالة للغاية	استخدام الذاكرة غير فعال.
غالبًا ما يظل التعقيد الزمني كما هو مع زيادة حجم البيانات.	يزداد التعقيد الزمني مع زيادة حجم البيانات.
مثال: **الأشجار، البيان**.	مثال: المصفوفة، المكدس، الرتل، القوائم المترابطة.

مقدمة

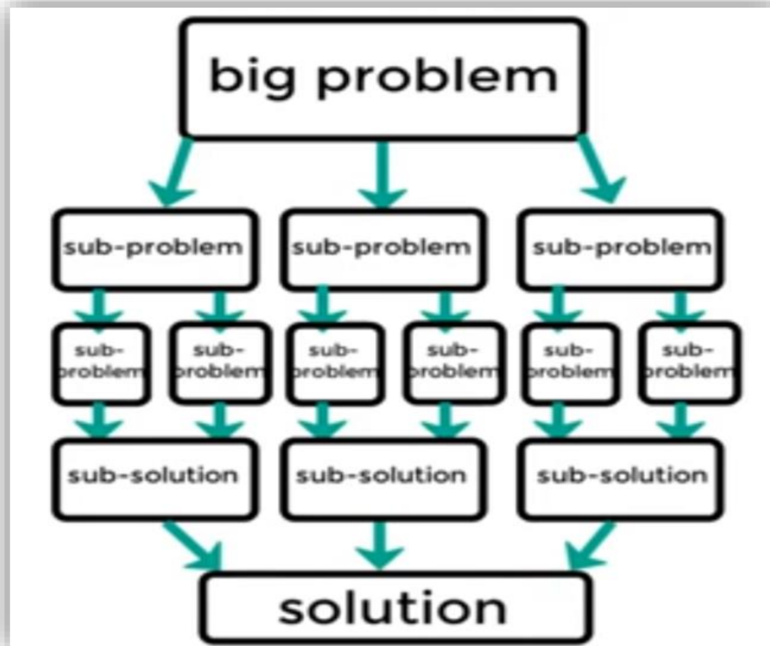
- الخوارزميات الطموحة أو الجشعة (Greedy Algorithms).
  - إحدى طرق تصميم الخوارزميات للمسائل الأمثلية.
    - ✓ تتمثل مسألة التحسين في إيجاد أفضل الحلول (أقل كلفة أو أعلى ربح) من جميع الحلول الممكنة.
  - تجد **الحل الأمثل الكلي** على **عدة مراحل**:
    - ✓ في كل مرحلة يتم ايجاد **الحل الامثل محليا** والبناء عليه في خطوات المسألة التالية.
  - لا يمكننا تغيير قرار متخذ في مرحلة سابقة، لذلك يمكن أن ينتج **حل مقبول** (feasible solution) للمسألة ولكن ليس **الحل الأمثل الكلي**.
  - امثلة:
    - ✓ مسألة الحقيبة Knapsack problem
    - ✓ مسألة توليد أطوال المسارات الأقصر بترتيب تصاعدي (خوارزمية Dijkstra)
    - ✓ مسألة ايجاد أشجار الإمتداد ذات الكلفة الأقل (Minimal Spanning Tree)
      - خوارزمية Kruskal و خوارزمية Prim.





مقدمة

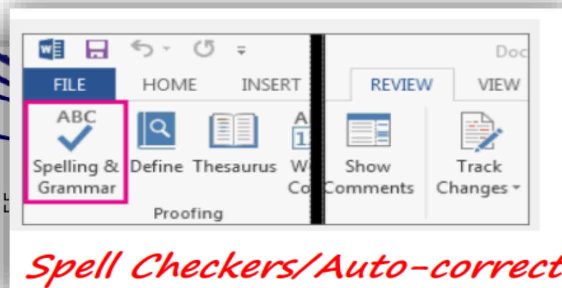
- تقنية تقسيم المسألة و تجميع الحلول أو تقنية فرق تسد (Divide and Conquer):
  - بفرض لدينا مسألة ما P حجم المعطيات فيها n ، تقترح هذه الإستراتيجية:
    - ✓ تقسيم المسألة P إلى k من المسائل الجزئية والتي تكون من **نوع المسألة الأصلية نفسه**.
      - إذا كانت بعض المسائل الجزئية لا تزال كبيرة – أي حجم المعطيات فيها لا يزال كبيراً – فيتم إعادة تطبيق الإستراتيجية **بأسلوب عودي** لتقسيم تلك المسائل إلى مسائل جزئية أصغر.
    - ✓ يتم حل المسائل الجزئية الأصغر
    - ✓ بعد ذلك يتم تجميع الحلول لتكوين حل المسألة الأصلية
  - امثلة:
    - ✓ خوارزمية الفرز السريع
    - ✓ وخوارزمية الفرز بالدمج



مقدمة



- البرمجة الديناميكية (Dynamic Programming):
  - ○ إحدى طرق تصميم الخوارزميات للمسائل الأمثلية.
  - ○ تشبه طريقة **التقسيم والتجميع**.
    - ✓ البرمجة الديناميكية **قابلة للتطبيق** عندما تكون المسائل الجزئية **غير مستقلة**.
  - ○ لحل مسألة ما،
    - ✓ يتم تقسيم المسألة إلى **مسائل جزئية**
    - ✓ ومن ثم ايجاد **الحلول المثلى الجزئية** **وتخزينها** وذلك لتوفير إعادة حسابها عند استخدامها.
    - ✓ يتم تجميع **الحلول المثلى الجزئية** من أجل ايجاد الحل الامثل **لاجزاء المسألة الأكبر** ومن ثم الحصول على **الحل الأمثل الكلي** .
  - ○ امثلة:
    - ✓ مسألة المسارات الأقصر (All Pairs Shortest Paths)
    - ✓ مسألة مسافة التحرير (Edit Distance)
    - ✓ مسألة البائع المتجول (Traveling Salesman Problem)
    - ✓ مسألة إيجاد أطول متتالية جزئية مشتركة (Longest Common Subsequence)

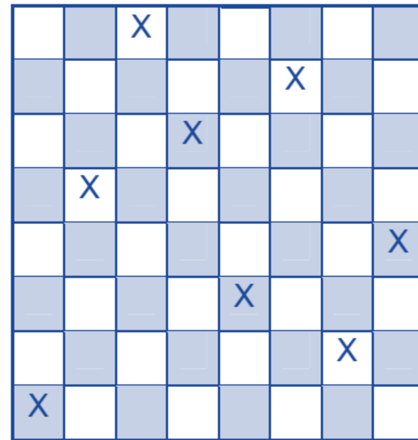


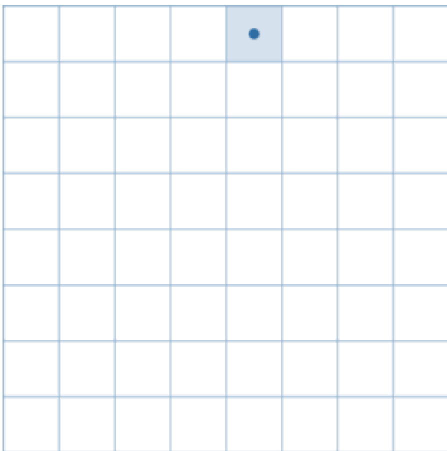
	A	B	A	Z	D	C
B	0	1	1	1	1	1
A	1	1	2	2	2	2
C	1	1	2	2	2	3
B	1	2	2	2	2	3
A	1	2	3	3	3	3
D	1	2	3	3	4	4

Bottom the top approach

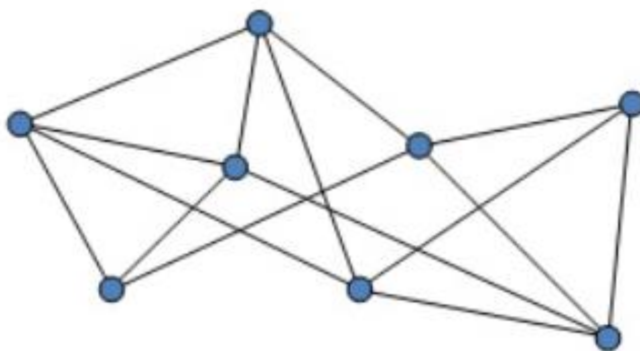
مقدمة

- **الخوارزميات التراجعية (Backtracking Algorithms):**
  - خوارزميات **عودية** تعتمد طريقة "التجريب والخطأ" (Try–and–Error) في حل المسائل.
  - تتلخص هذه الطريقة ببناء الحل النهائي للمسألة عن طريق مجموعة من الخطوات:
    - ✓ في كل خطوة نحدد **الإمكانيات المتاحة** للخطوة التالية.
    - ✓ ندرس هذه الإمكانيات لكي نختار أحدها **ونسجلها** على أنها الخطوة التالية في الحل النهائي.
    - ✓ نتابع الخوارزمية اعتماداً على الخطوة **السابقة المختارة**.
    - ✓ عندما يتأكد لنا أن اختيارنا **لا يقود** إلى الحل النهائي أو يؤدي إلى طريق مسدود، نتراجع عن الخطوة المختارة.
  - **امثلة:**
    - ✓ مسألة جولة حصان الشطرنج
    - ✓ مسألة الوزراء الثمانية
    - ✓ لمسألة الحقيبة (Knapsack Problem).





البيان (graph)



البيان (graph)

- بنية معطيات **غير خطية**.
- نرمز له كثنائية مرتبة:

G=<V,E>

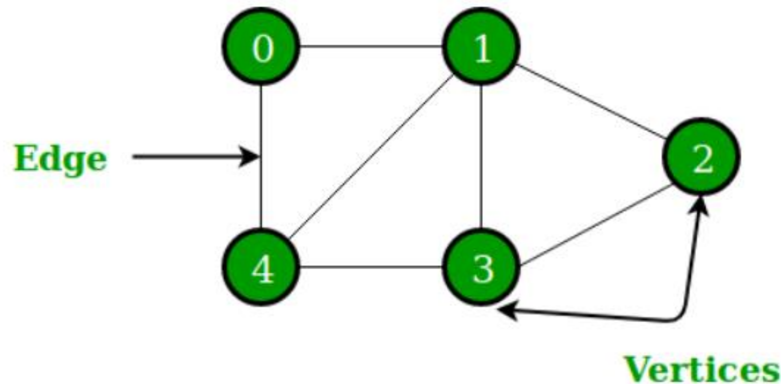


  - V مجموعة منتهية و**غير خالية** من العقد أو ما يدعى بالرؤوس (vertices).



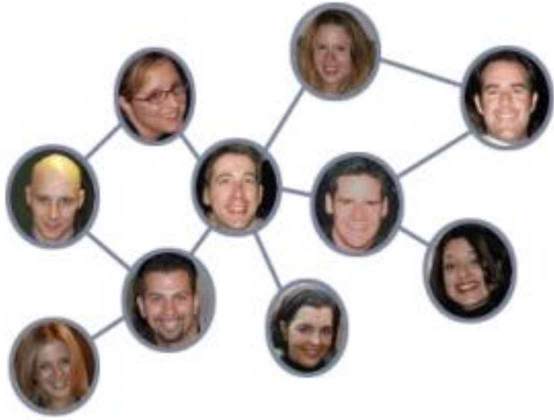
  - E مجموعة منتهية من الاضلاع أو ما يدعى بالحافات (edges) والتي بدورها تصل الرؤوس مع بعضها.

- من أجل بيان محدد G يرمز لـ:
  - مجموعة الرؤوس فيه بـ V(G).
  - مجموعة الأضلاع فيه بـ E(G).



البيان (graph)

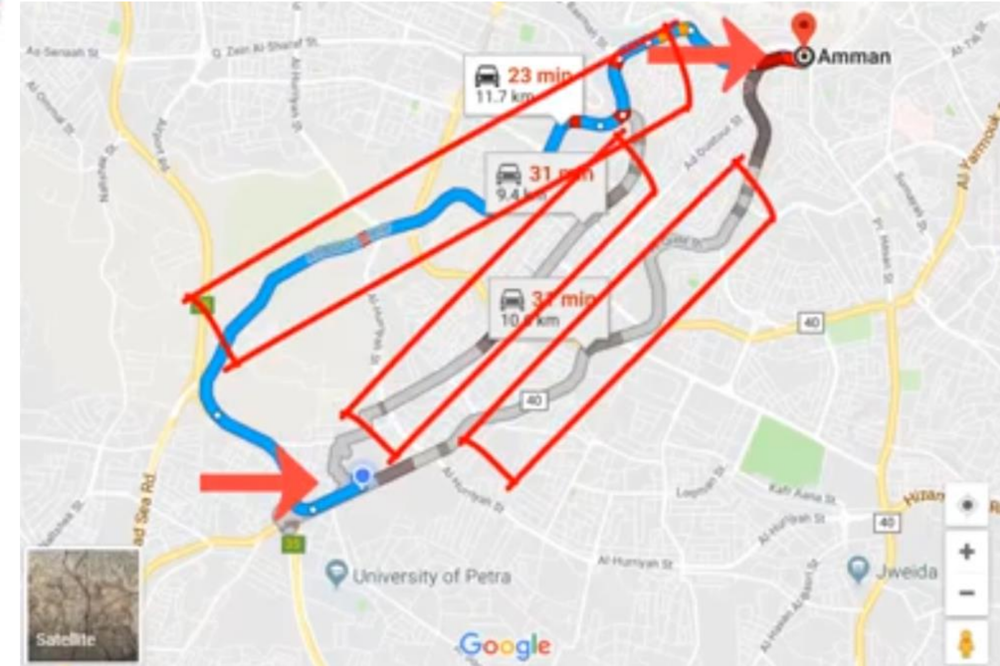
- من اشهر تطبيقات البيان:



Social Networks



World Wide Web



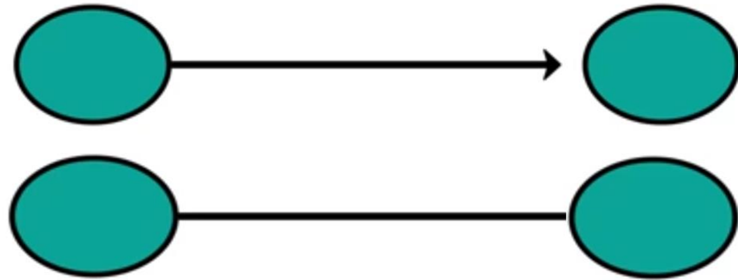
طلبات الصداقة
بحث عن أصدقاء · الإعدادات
أشخاص قد تعرفهم
إضافة صديق
إزالة
إضافة صديق
إزالة

من أهم منهجيات حل المسائل في الذكاء الصنعي:

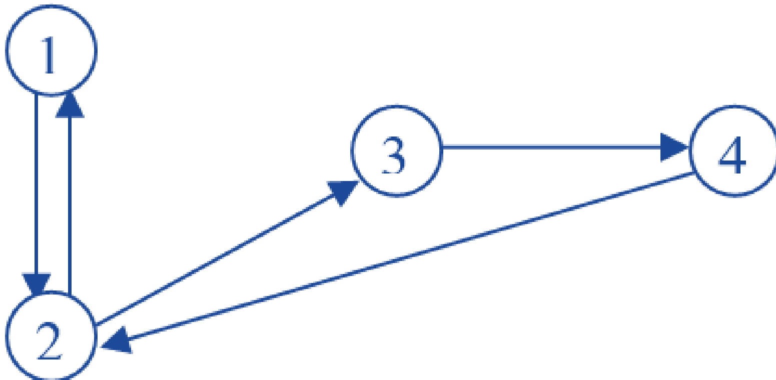
– تحويل المسألة إلى مسألة بحث في بيان

البيان (graph)

- في البيان نميز بين:
  - البيان الموجه (directed graph).
  - البيان غير الموجه (undirected graph).

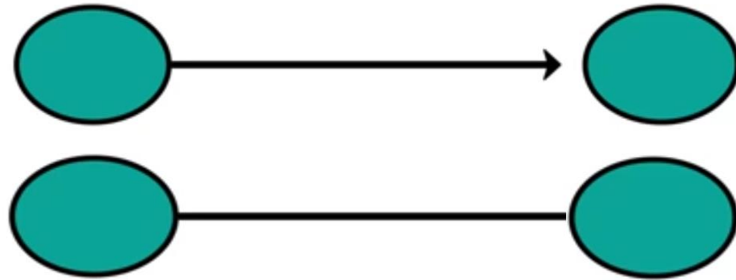


1. **البيان الموجه (directed graph).**
   - يمثل كل ضلع فيه <u>**بالثنائية المرتبة**</u> <u,v> والتي بدورها تحدد **وجود ضلع** من الرأس u إلى الرأس v.
   - حسب التعريف السابق، <v,u> ,<u,v> عبارة عن **ضلعين مختلفين** في البيان الموجه.
   - **مثال:** ليكن البيان الموجه التالي G1



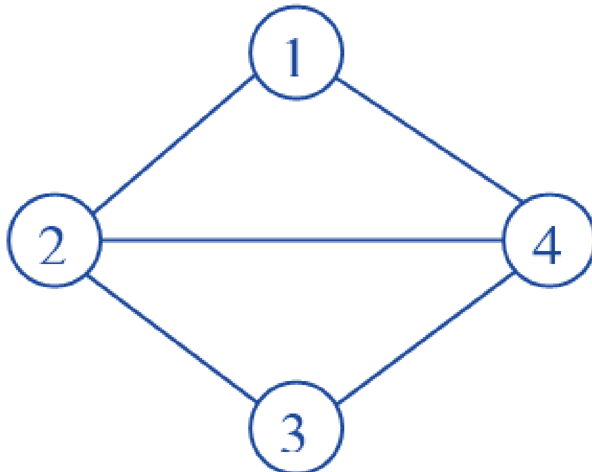
$$V(G1) = \{1,2,3,4\}$$
$$E(G1) = \{\langle 1,2\rangle,\langle 2,1\rangle,\langle 2,3\rangle,\langle 3,4\rangle,\langle 4,2\rangle\}$$

البيان (graph)



- في البيان نميز بين:
  - البيان الموجه (directed graph).
  - البيان غير الموجه (undirected graph).

2. **البيان غير الموجه (undirected graph).**
   - يمثل كل ضلع فيه **بالثنائية غير المرتبة** (u, v) والتي بدورها تحدد وجود **علاقة متبادلة** بين الرأس u والرأس v.
   - حسب التعريف السابق، الضلع (u,v) هو **نفس** الضلع (v,u) في البيان غير الموجه.
   - **مثال:** ليكن البيان غير الموجه التالي G2:



V(G2) = {1,2,3,4}
E(G2) = {(1,2),(1,4),(2,3),(2,4),(3,4)}

البيان (graph)

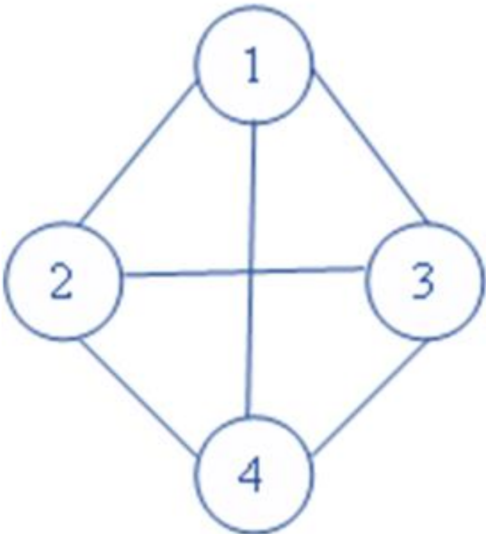
- **ملاحظة(1)**: العدد الأعظمي للأضلاع في **بيان موجه** يتضمن n رأساً يساوي

$$n(n-1)$$

- **ملاحظة(2)**: العدد الأعظمي للأضلاع في **بيان غير موجه** يتضمن n رأساً يساوي

$$\frac{n(n-1)}{2}$$

  - مثال: بالنسبة لبيان غير موجه يتضمن 4 رؤوس، يكون العدد الأعظمي للأضلاع فيه ($\frac{4(4-1)}{2}$=6).



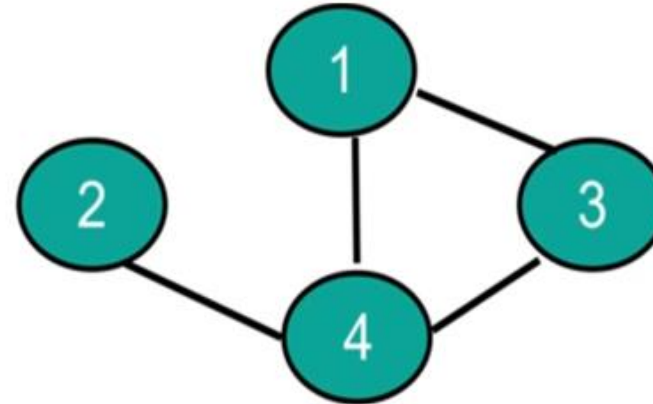
تعاريف أساسية

- البيان الكامل – البيان الجزئي
- المسار (path)
  - المسار البسيط (simple path)
  - الحلقة (cycle)
- البيان الحلقي – البيان غير الحلقي
- الرؤوس المتجاورة – الرؤوس المرتبطة
- البيان المترابط (Connected Graph)
- المركبة المترابطة
- درجة رأس (Node Degree)

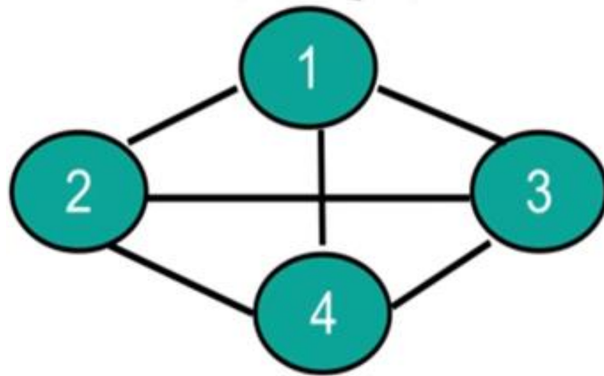
البيان الكامل (complete graph)

- هو البيان الذي يمكن فيه لأي عقدة **الوصول** إلى أي عقدة أخرى **بشكل مباشر**.
  - أي كل عقدة **مرتبطة مباشرة** مع جميع العقد الاخرى← هو بيان يتضمن **العدد الأعظمي** من الأضلاع.

- مثال:



G2
بيان غير كامل



G1
بيان كامل

  - وبالتالي البيان G1، هو بيان كامل لأنه يتضمن 6 أضلاع بينما البيان G2 هو بيان غير الكامل لأنه يتضمن 4 أضلاع فقط.

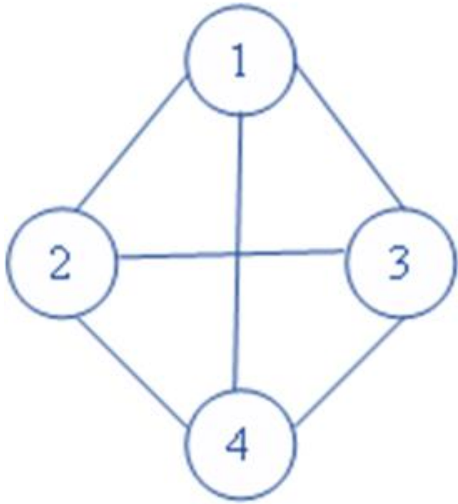
البيان الجزئي (sub-graph)

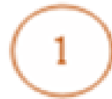
- البيان الجزئي من G هو بيان G' يحقق:

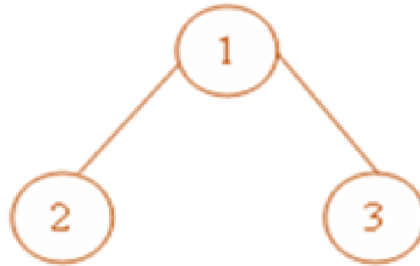
$$V(G') \subseteq V(G)$$
$$E(G') \subseteq E(G)$$

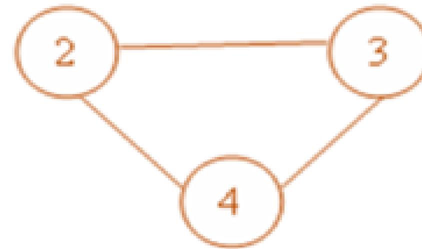
- مثال: يوضح الشكل التالي بعض البيانات الجزئية من البيان G_1.

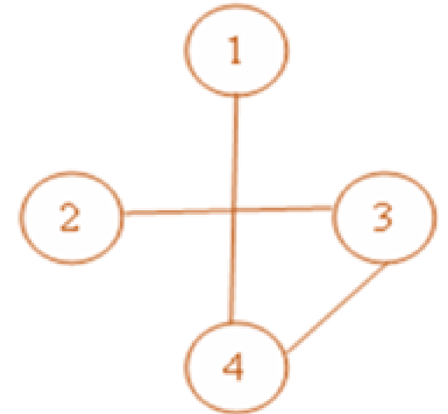
G1











V(G1) = {1,2,3,4}

E(G1) = {(1,2),(1,3),(1,4),(2,3),(2,4),(3,4)}

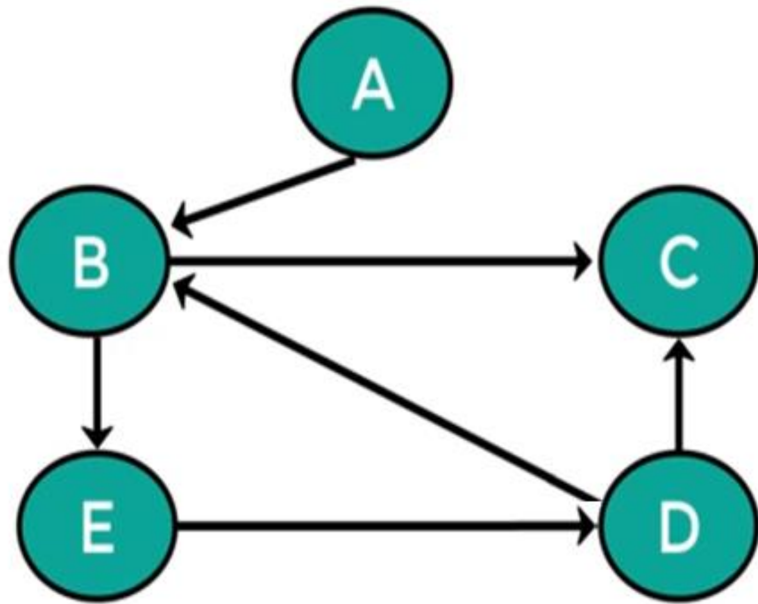
المسار (path)

- المسار من الرأس u إلى الرأس v في البيان G هو متتالية من الرؤوس

$$\boldsymbol{u,i_1,i_2,\ldots,i_k,v}$$

بحيث :

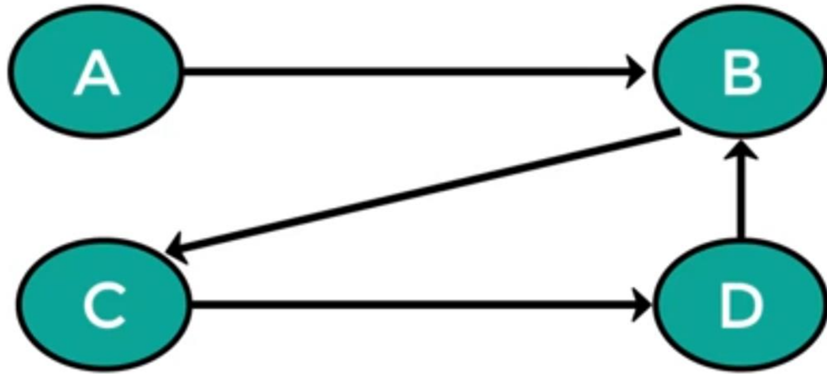
  - في حالة البيان **غير الموجه**: يكون لدينا $(u,i_1),(i_1,i_2),\ldots,(i_k,v)$ في E(G).
  - في حالة البيان **الموجه**: يكون لدينا $< u,i_1 > , < i_1\ ,i_2 > , \ldots , < i_k\ ,v >$ في E(G).



- طول المسار يساوي إلى **عدد أضلاع** المسار.

المسار (path)

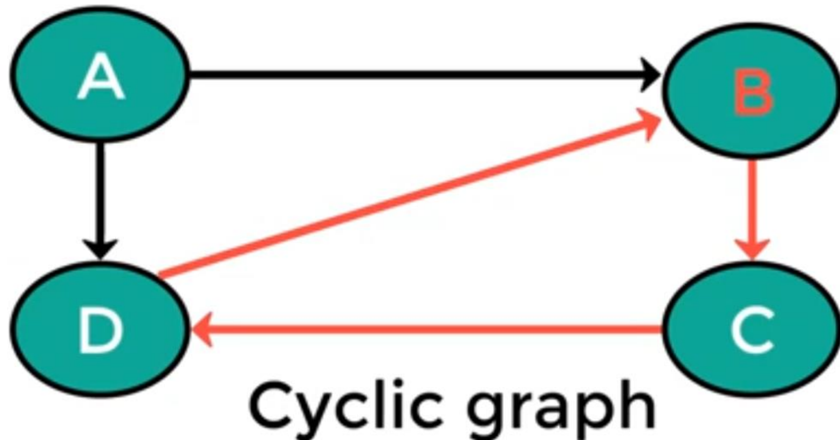
- المسار البسيط (simple path): هو المسار الذي تكون فيه كل الرؤوس **مختلفة** .
  - مثال:



path : A,B,C,D,B not simple

path : A,B,C,D simple path

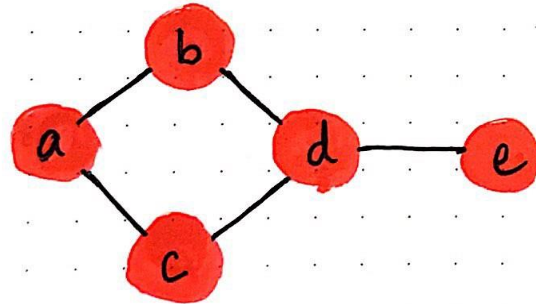
- الحلقة (cycle): مسار بسيط فيه الرأس الأول هو نفسه الرأس الأخير.



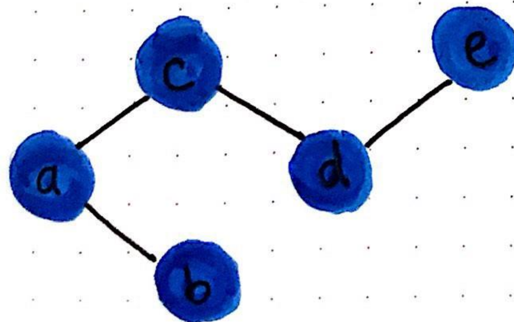
path : B,C,D,B

البيان الحلقي وغير الحلقي

- نقول عن بيان أنه
  - حلقي (Cyclic) أذا كان يتضمن مسار ما يبدأ من عقدة ما و ينتهي عند نفس القعدة (اي يتضمن حلقة).
  - غير حلقي (acyclic) أذا كان لا يتضمن أي حلقة.



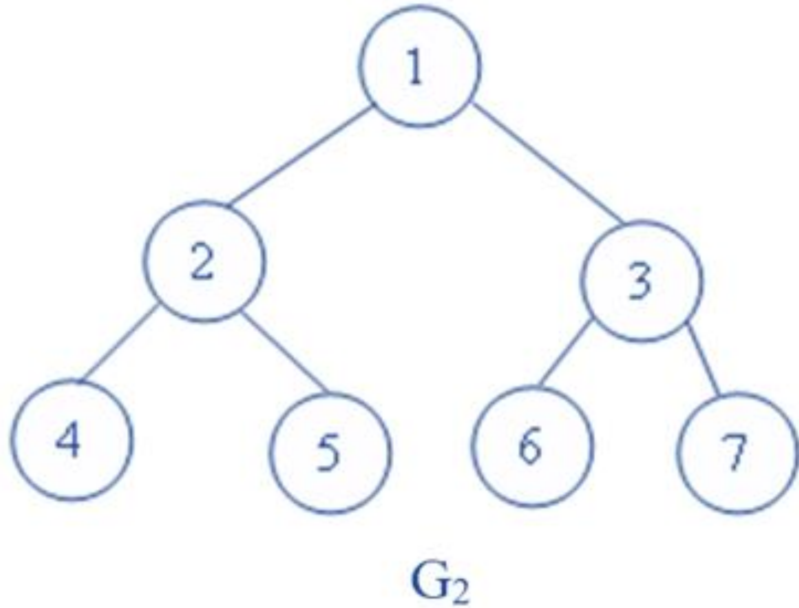
A graph with at least one cycle is known as a cyclic graph.



a graph with no cycles in it is known as an acyclic graph.

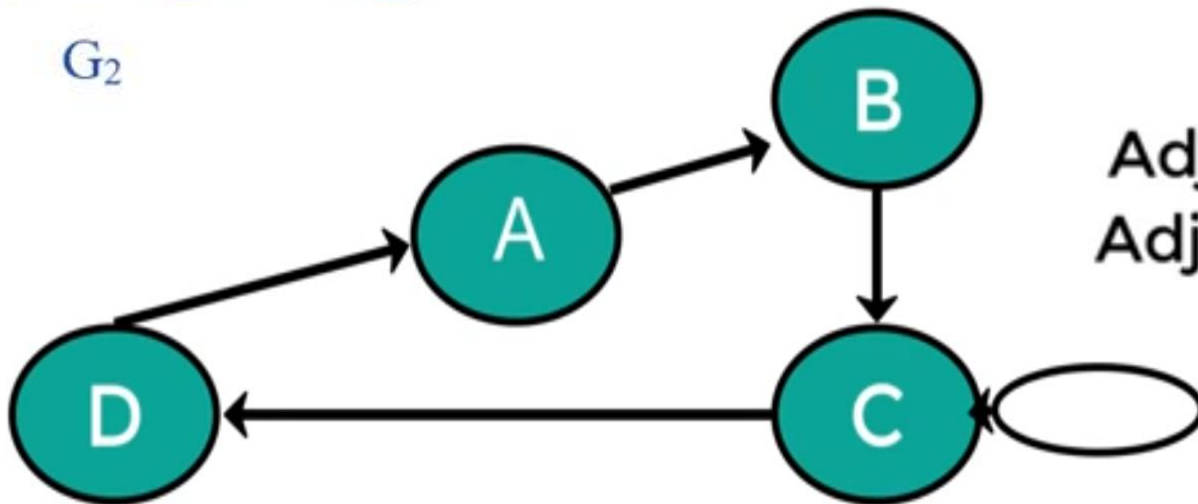
الرؤوس المتجاورة (adjacent vertices)

- نقول عن الرأسين u و v في بيان G إنهما متجاوران إذا وفقط إذا **وجد ضلع** من الرأس u إلى الرأس v (أي أن الرأسين u و v مرتبطين **بشكل مباشر**).
- **مثال (1):** بفرض لدينا البيان غير الموجه G2
  - ماهي الرؤوس المجاورة للرأس (2)؟



{1,4,5}

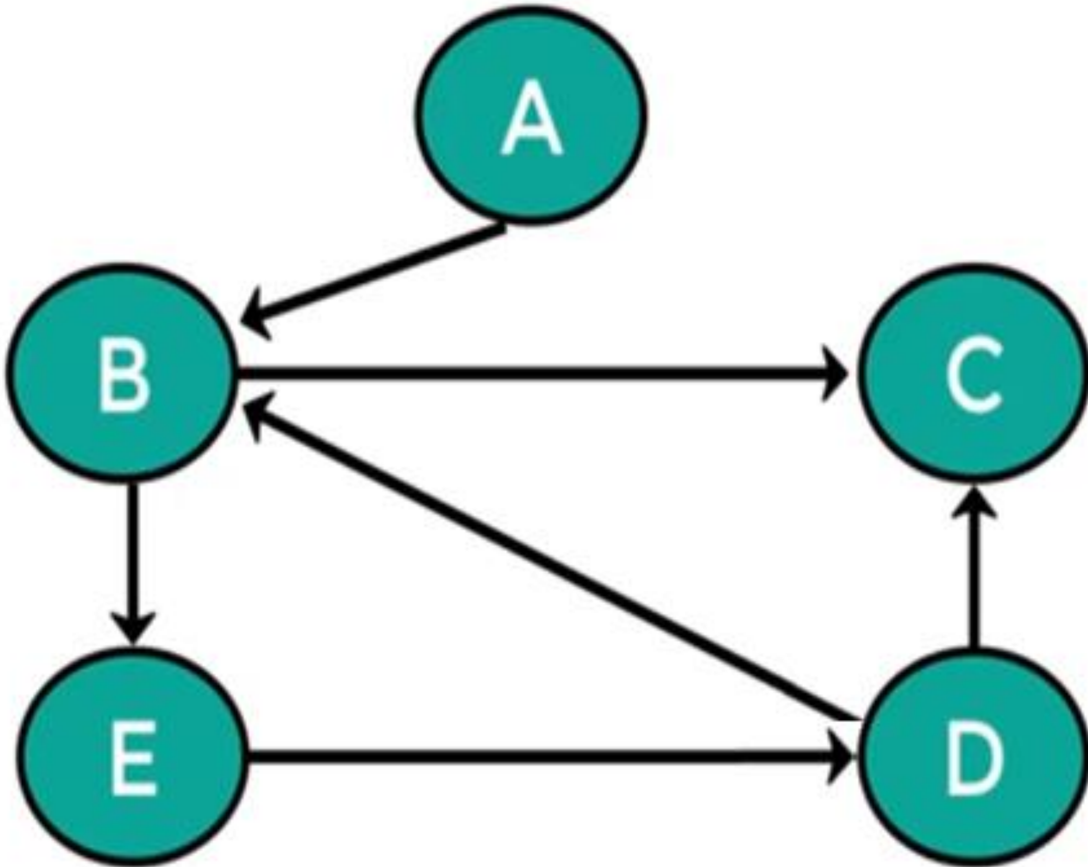
- **مثال (2):** بفرض لدينا البيان الموجه التالي:



Adjacent Vertices(A): {B}
Adjacent Vertices(C):{D,C}

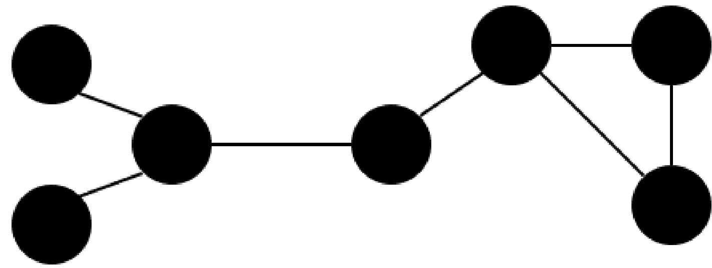
الرؤوس المرتبطة (connected vertices)

- نقول عن الرأسين u و v في بيان G إنهما مرتبطان (أو متصلان) إذا وفقط إذا **وجد مسار** من الرأس u إلى الرأسv.
- **مثال:** الرأس A مرتبط مع الرأسD بينما العكس غير صحيح.

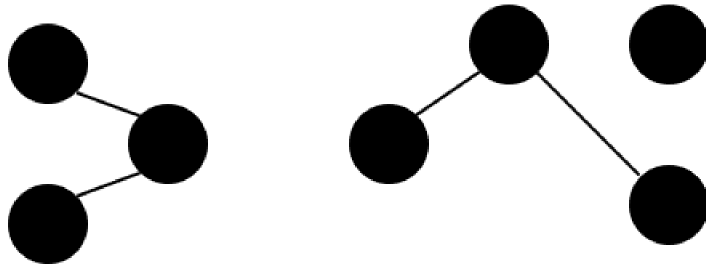


البيان المترابط (Connected Graph)

- نقول عن **بيان G** إنه **مترابط (متصل)** إذا وفقط إذا كان **كل رأسين فيه مرتبطين** (أي هناك مسار بين كل رأس وكل الرؤوس الاخرى).
- **مثال (1)**:بفرض لدينا البيانات غير الموجهة التالية

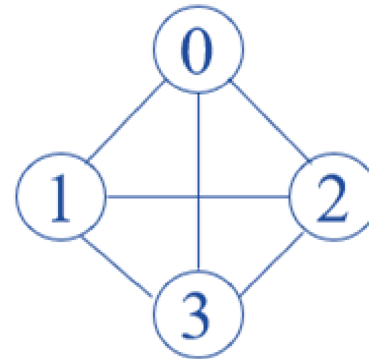


Connected graph

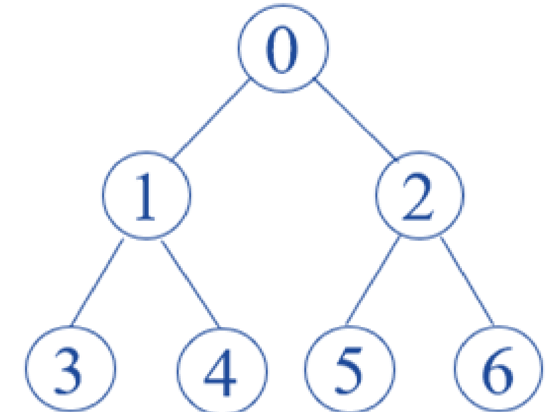


Disconnected graph

connected



G_1

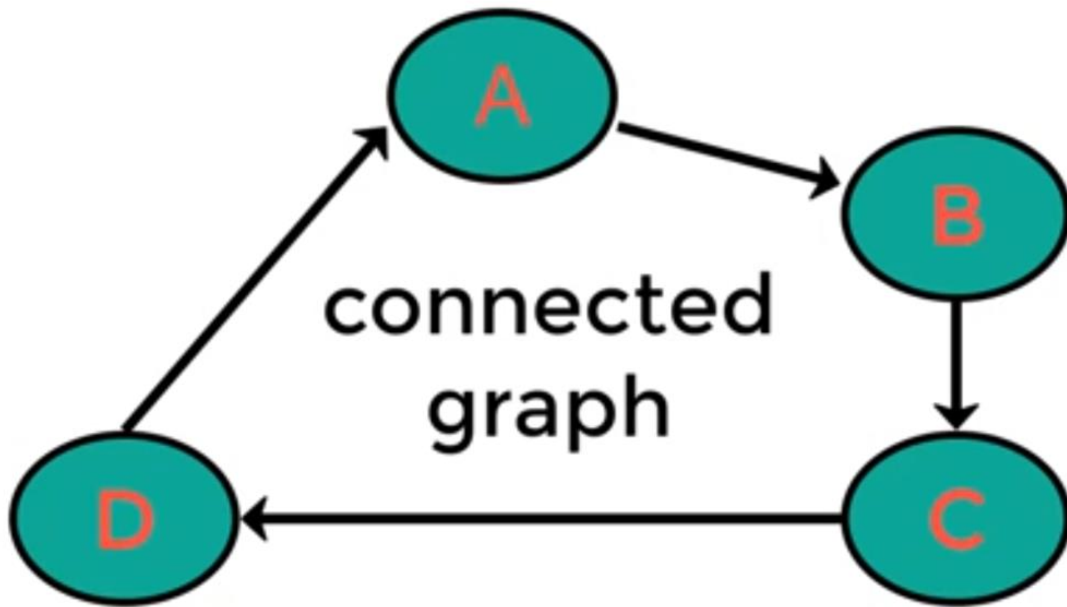


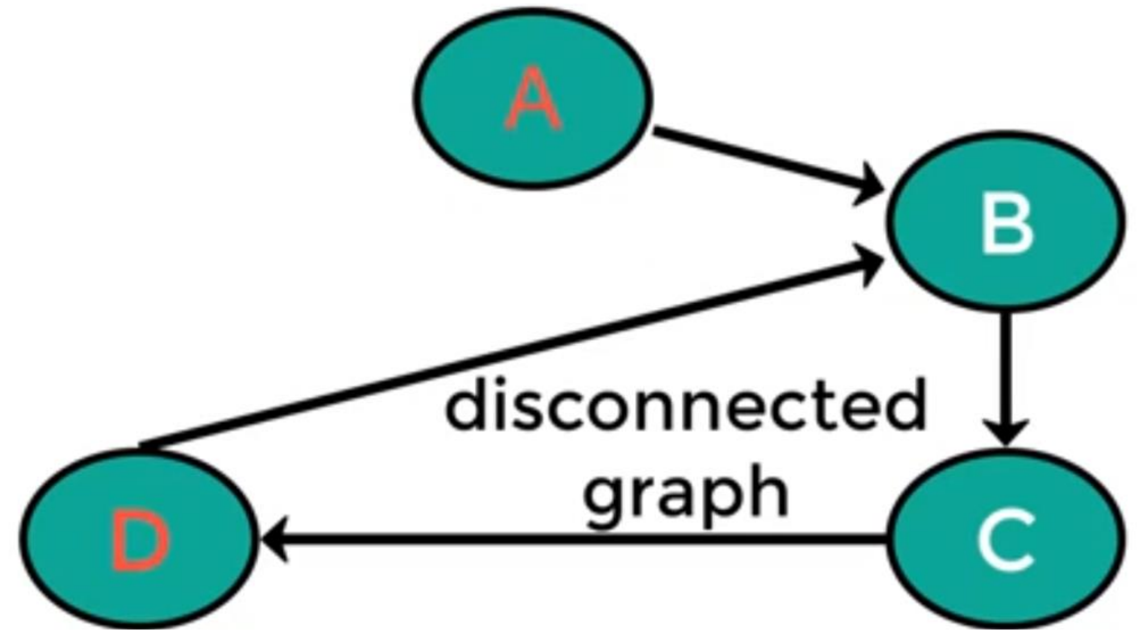
G_2

الشجرة هي بيان مترابط لا يتضمن أي حلقة

البيان المترابط (Connected Graph)

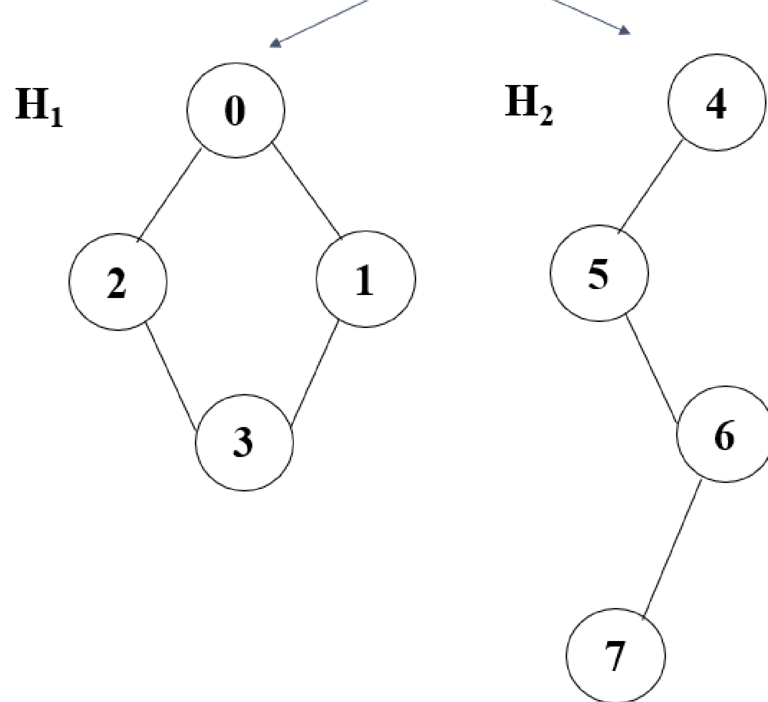
- نقول عن **بيان G** إنه **مترابط (متصل)** إذا وفقط إذا كان **كل رأسين فيه مرتبطين** (أي هناك مسار بين كل رأس وكل الرؤوس الاخرى).
- **مثال (2)**:بفرض لدينا البيانات الموجهة التالية



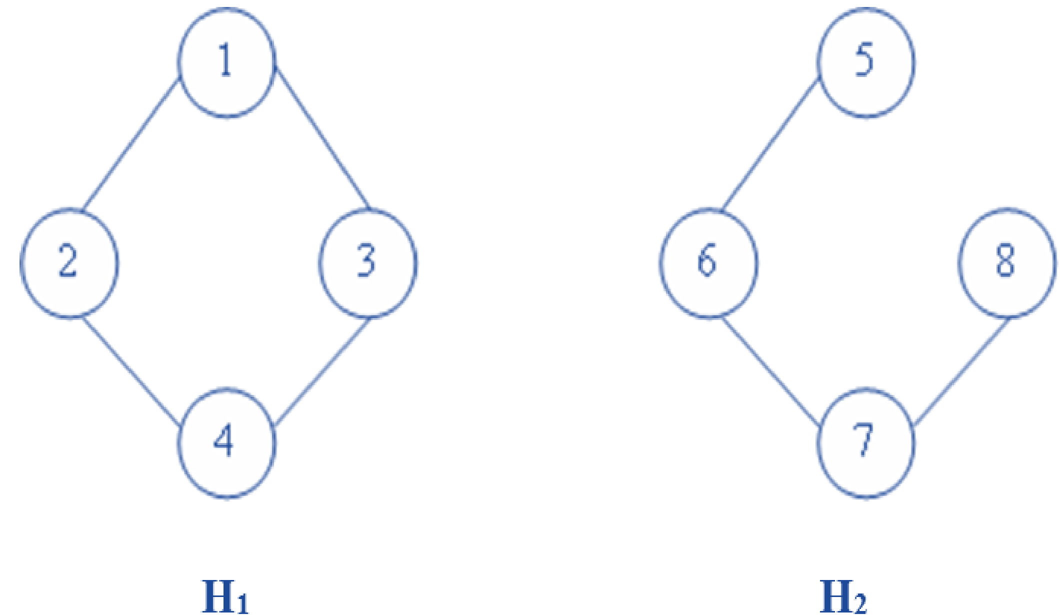


المركبة المترابطة (connected component)

- المركبة المترابطة H من بيان G هي **أكبر بيان جزئي مترابط** من البيان G .
  - لا يمكن إيجاد بيان جزئي آخر من البيان G مترابط ويتضمن H .
- **مثال(1):** يوضح الشكل التالي بيانات غير موجهة وغير مترابطة (disconnected) والمركبات المترابطة الموافقة لها.



G_4 (not connected)



بيان غير موجه G_4 يتضمن مركبتين

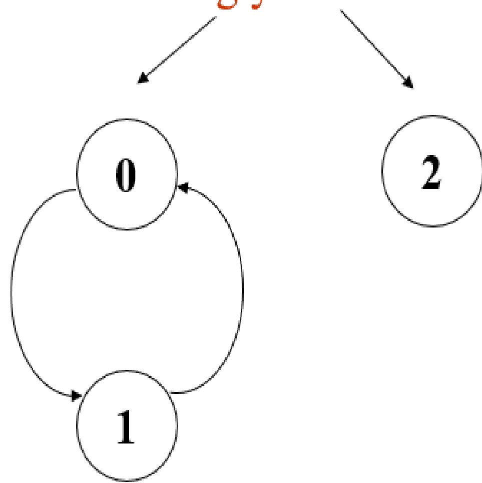
المركبة المترابطة (connected component)

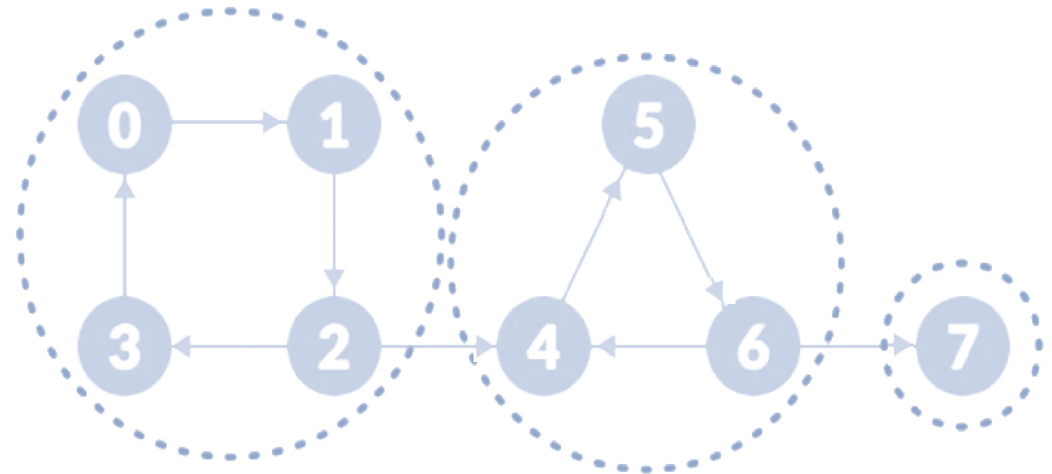
- المركبة المترابطة H من بيان G هي **أكبر بيان جزئي مترابط** من البيان G .
  - لا يمكن إيجاد بيان جزئي آخر من البيان G مترابط ويتضمن H .
- **مثال(2)**: يوضح الشكل التالي بيانات موجهة وغير مترابطة (disconnected) والمركبات المترابطة الموافقة لها.

not strongly connected



G3



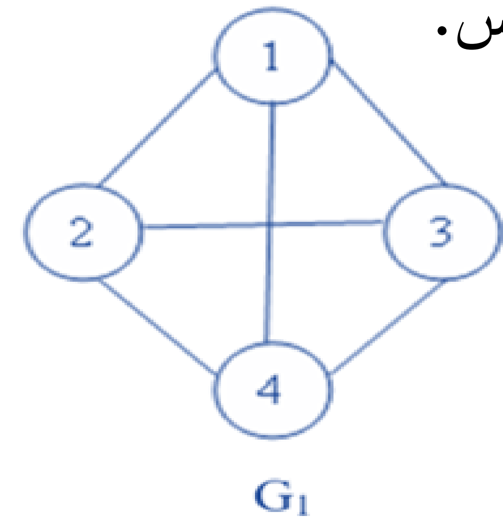


Strongly connected components

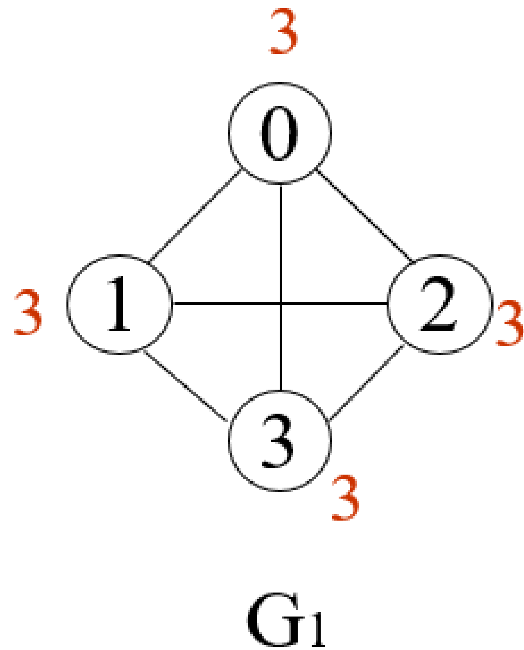
درجة رأس (Node Degree)

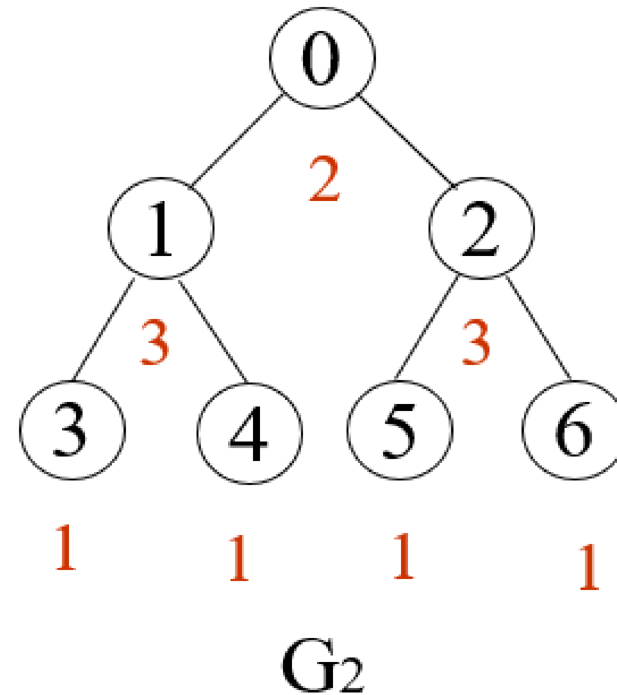
- في بيان **غير موجه**، نعرف درجة رأس (degree) بأنها عدد الأضلاع المارة بهذا الرأس.
  - **مثال1**: درجة الرأس (1) في البيان غير الموجه G1 :

degree(1)=3



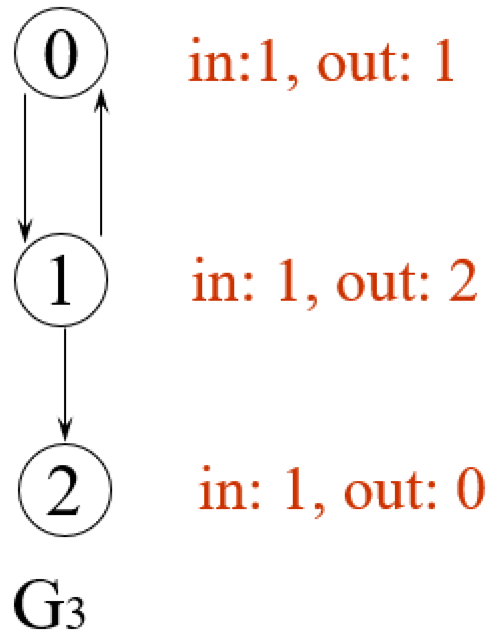
  - **مثال 2**:

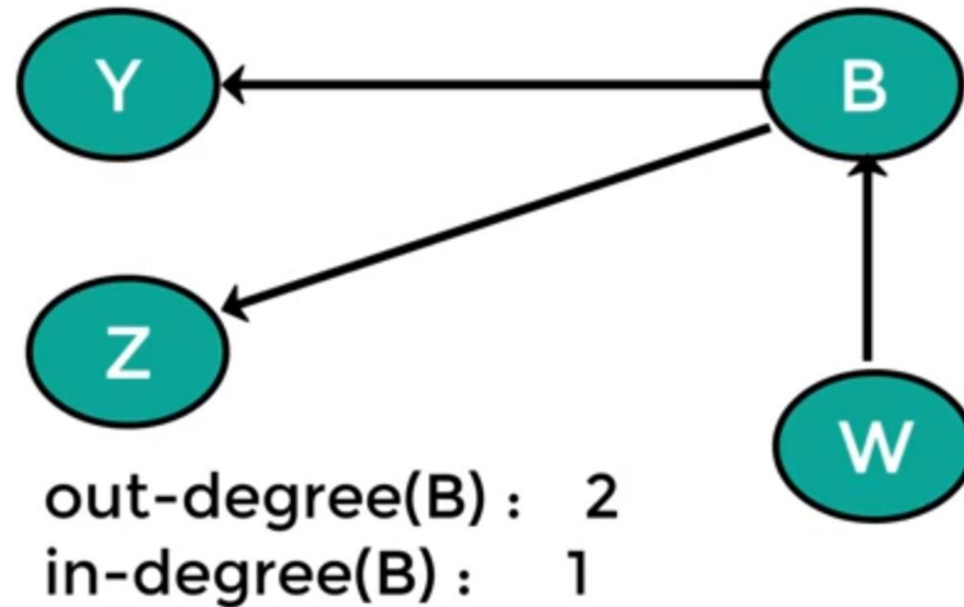




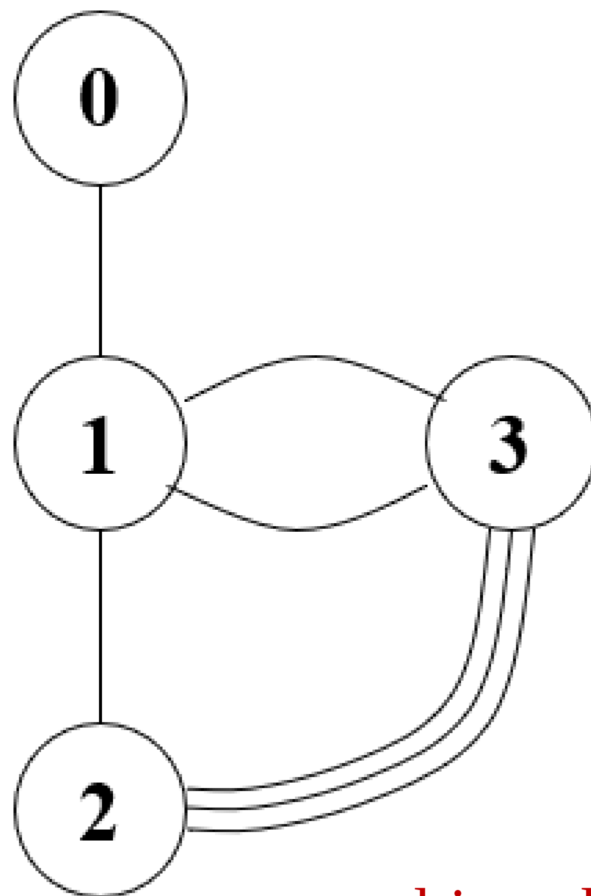
درجة رأس (Node Degree)

- في البيان **الموجه**، نعرف
  - **الدرجة الواردة** in–degree للرأس v بأنها عدد الأضلاع **القادمة** لـ v.
  - **الدرجة الصادرة** out–degree للرأس v بأنها عدد الأضلاع **الخارجة** من v.
- **مثال :**





Multigraph



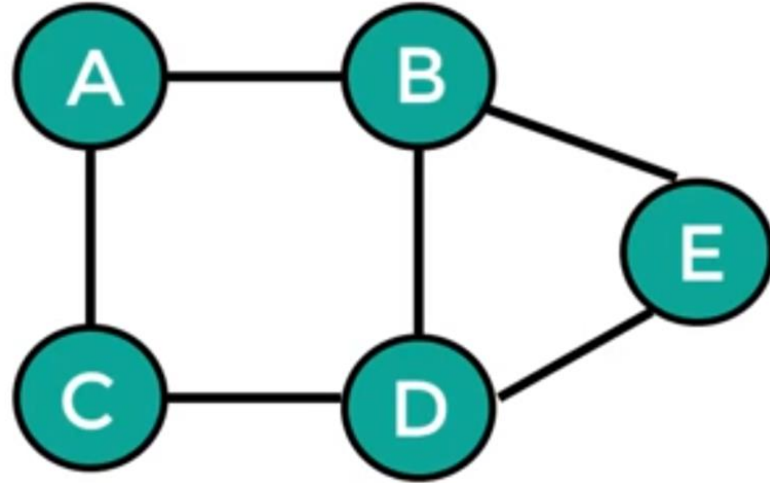
multigraph:
multiple occurrences of the same edge

تمثيل البيان

- يمكن تمثيل البيان بطرق عديدة نذكر منها:
  - تمثيل البيان باستخدام **مصفوفة التجاور** Adjacency Matrix
  - تمثيل البيان باستخدام **قوائم التجاور** Adjacency Lists

- **ملاحظة:** يعتمد اختيار **التمثيل المناسب** للبيان على **نوع العمليات** التي سيتم إجراؤها وسهولة الاستخدام.

تمثيل البيان باستخدام مصفوفة التجاور

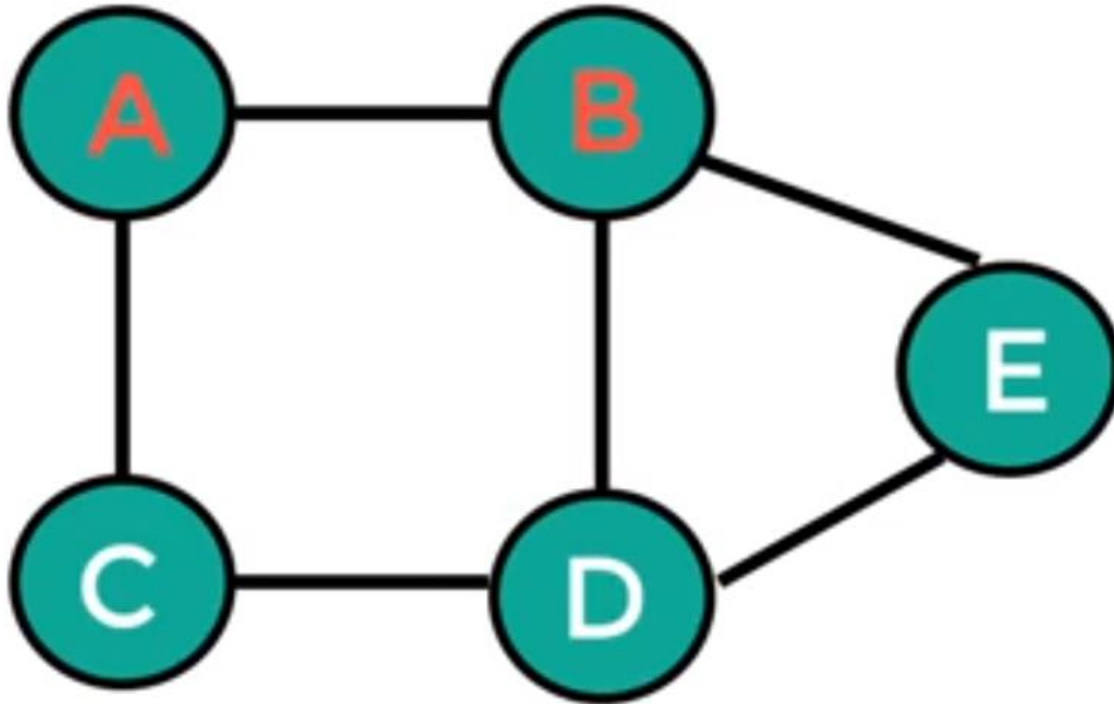


- مصفوفة التجاور للبيان G هي:
  - مصفوفة **مربعة** $A[n,n]$ حيث n عدد الرؤوس في البيان G.
  - الأسطر والأعمدة موافقة لرؤوس البيان.
  - عناصرها $A[i][j]$ تعطى بالصيغة الآتية:

$$A[i][j] = \begin{cases} 1 & \text{if there is an edge from vertex } i \text{ to vertex } j \\ 0 & \text{otherwise} \end{cases}$$

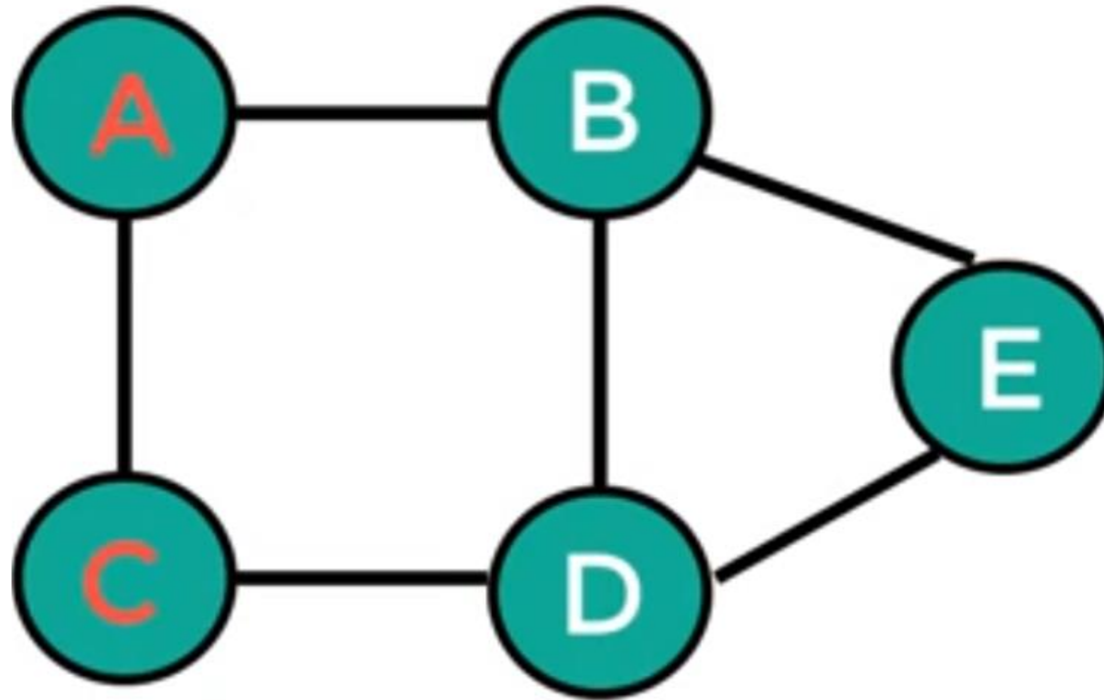
	A	B	C	D	E
A					
B					
C					
D					
E					

تمثيل البيان باستخدام مصفوفة التجاور



	A	B	C	D	E
A		1			
B	1				
C					
D					
E					

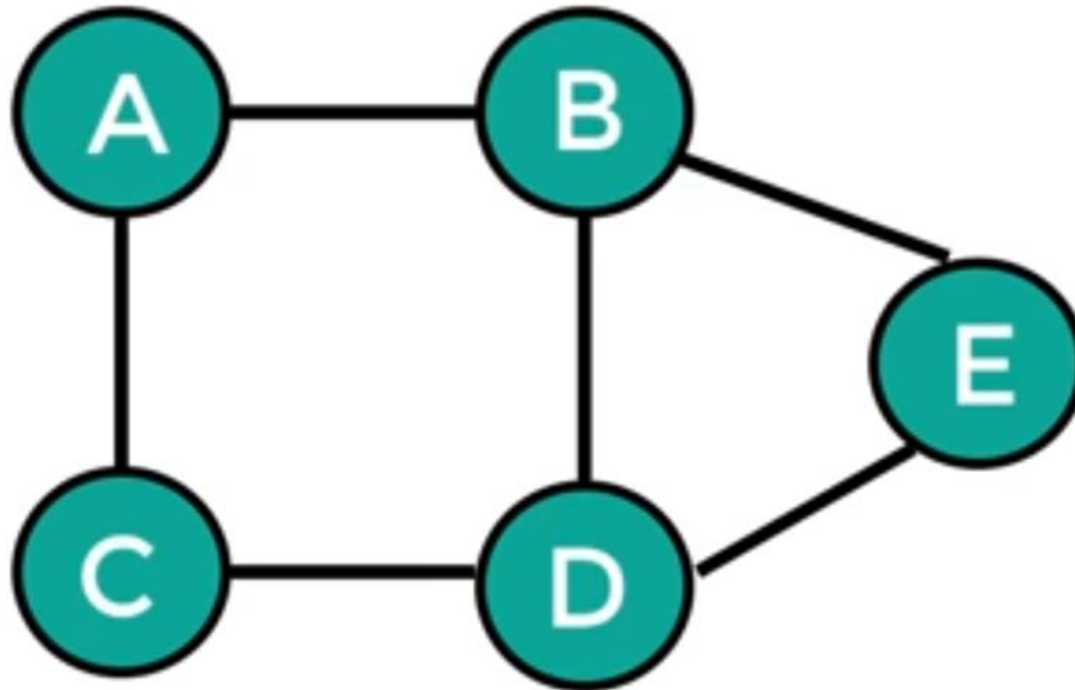
تمثيل البيان باستخدام مصفوفة التجاور



	A	B	C	D	E
A		1	1		
B	1				
C	1				
D					
E					

```
// to add edge into the matrix
  void add_edge(int u, int v)
   {    A[u][v] = 1;
        A[v][u] = 1; //just for undirected graph
   }
```

تمثيل البيان باستخدام مصفوفة التجاور

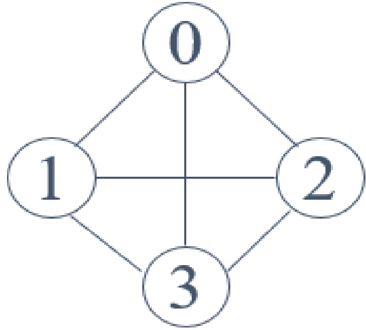


	A	B	C	D	E
A	0	1	1	0	0
B	1	0	0	1	1
C	1	0	0	1	0
D	0	1	1	0	1
E	0	1	0	1	0

- نلاحظ بأن في حالة البيان غير الموجه، تصبح هذه المصفوفة **متناظرة (symmetric)** بالنسبة إلى القطر الرئيسي .

تمثيل البيان باستخدام مصفوفة التجاور

- أمثلة:



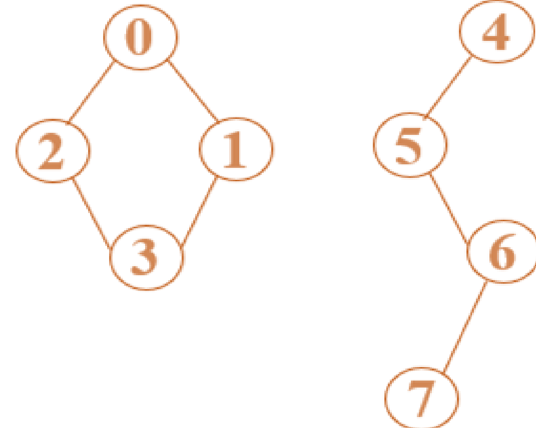
0	1	1	1
1	0	1	1
1	1	0	1
1	1	1	0

G_1



0	1	0
1	0	1
0	0	0

G_2



0	1	1	0	0	0	0	0
1	0	0	1	0	0	0	0
1	0	0	1	0	0	0	0
0	1	1	0	0	0	0	0
0	0	0	0	0	1	0	0
0	0	0	0	1	0	1	0
0	0	0	0	0	1	0	1
0	0	0	0	0	0	1	0

G_4

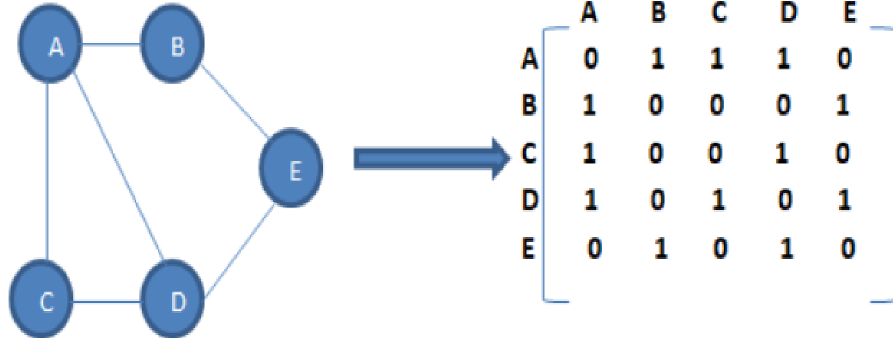
لحساب درجة رأس باستخدام مصفوفة التجاور

- نميز بين حالتين:

1. إذا كانت A مصفوفة التجاور **لبيان غير موجه** يكون:

$$degree(i) = \sum_{j=1}^{n} a[i][j]$$

مجموع عناصر السطر الموافق للرأس i



	A	B	C	D	E
A	0	1	1	1	0
B	1	0	0	0	1
C	1	0	0	1	0
D	1	0	1	0	1
E	0	1	0	1	0

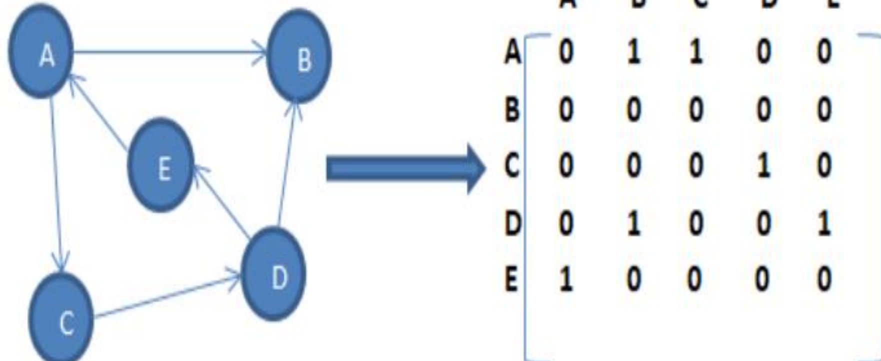
2. إذا كانت A مصفوفة التجاور **لبيان موجه** يكون:

$$out-degree(i) = \sum_{j=1}^{n} a[i][j]$$

مجموع عناصر السطر الموافق للرأس i

$$in-degree(j) = \sum_{i=1}^{n} a[i][j]$$

مجموع عناصر العمود الموافق للرأس j

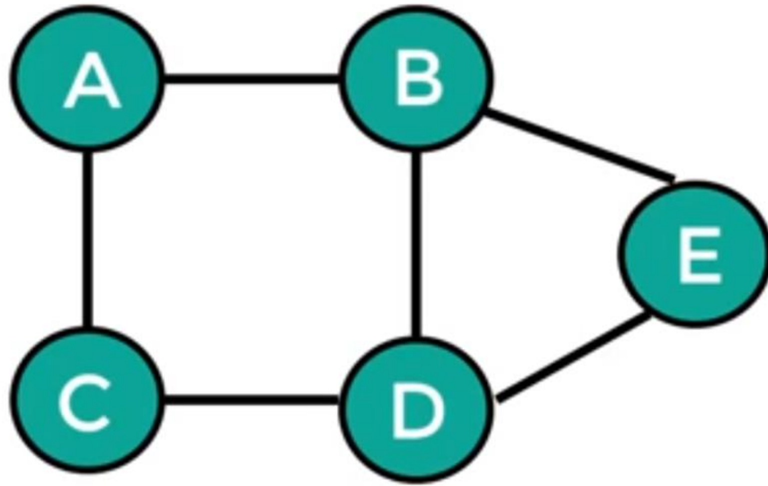


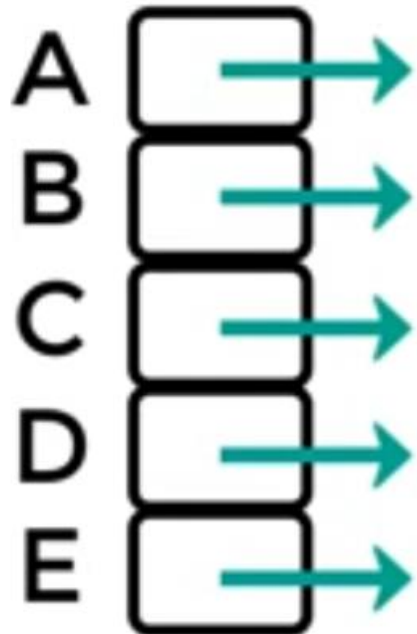
	A	B	C	D	E
A	0	1	1	0	0
B	0	0	0	0	0
C	0	0	0	1	0
D	0	1	0	0	1
E	1	0	0	0	0

تمثيل البيان باستخدام مصفوفة التجاور

- **الإيجابيات:**
  - سهولة التحقق.
  - الوصول سريع جداً وفعال من مرتبة O(1)
    - ✓ على سبيل المثال لمعرفة إذا كان هناك ضلع من الرأس u إلى الرأس v.
- **السلبيات:**
  - تستهلك مساحة تخزينية أكبر $O(n^2)$
    - ✓ حتى لوكان البيان sparse (أي يتضمن أقل عدد من الاضلاع)، يتم استهلاك نفس المساحة التخزينية.

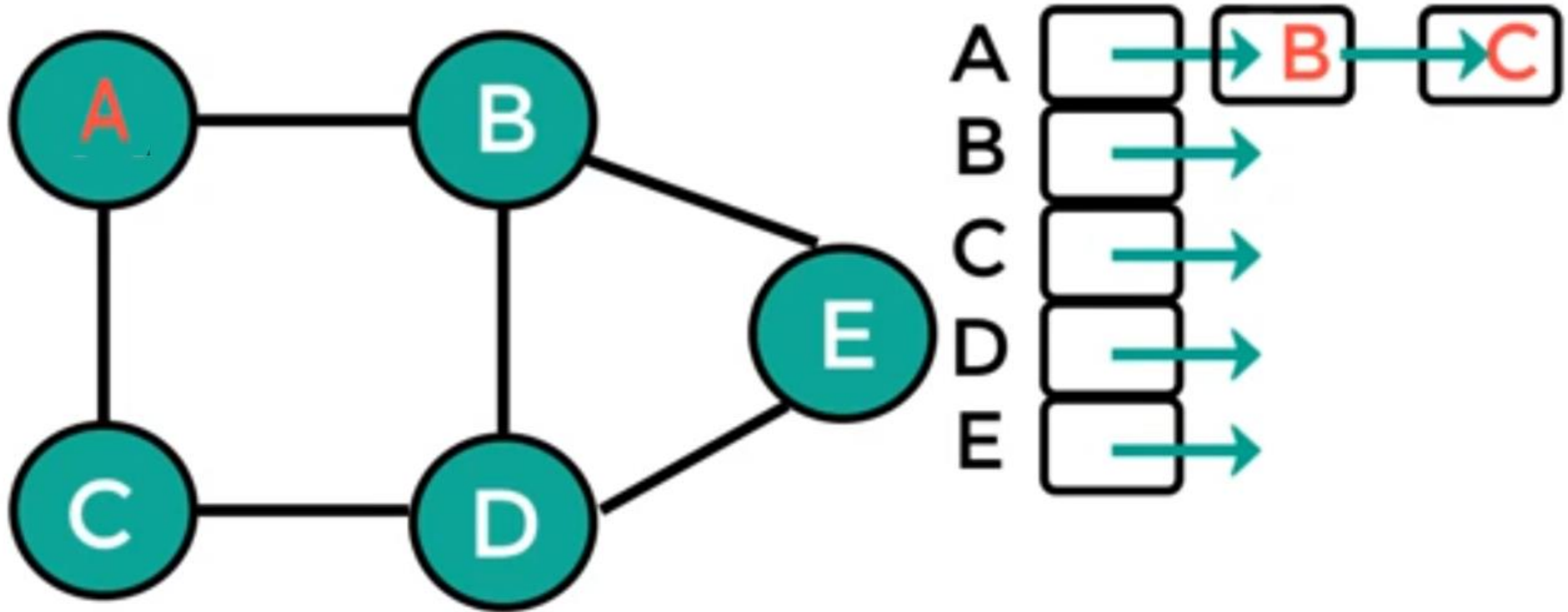
تمثيل البيان باستخدام قوائم التجاور



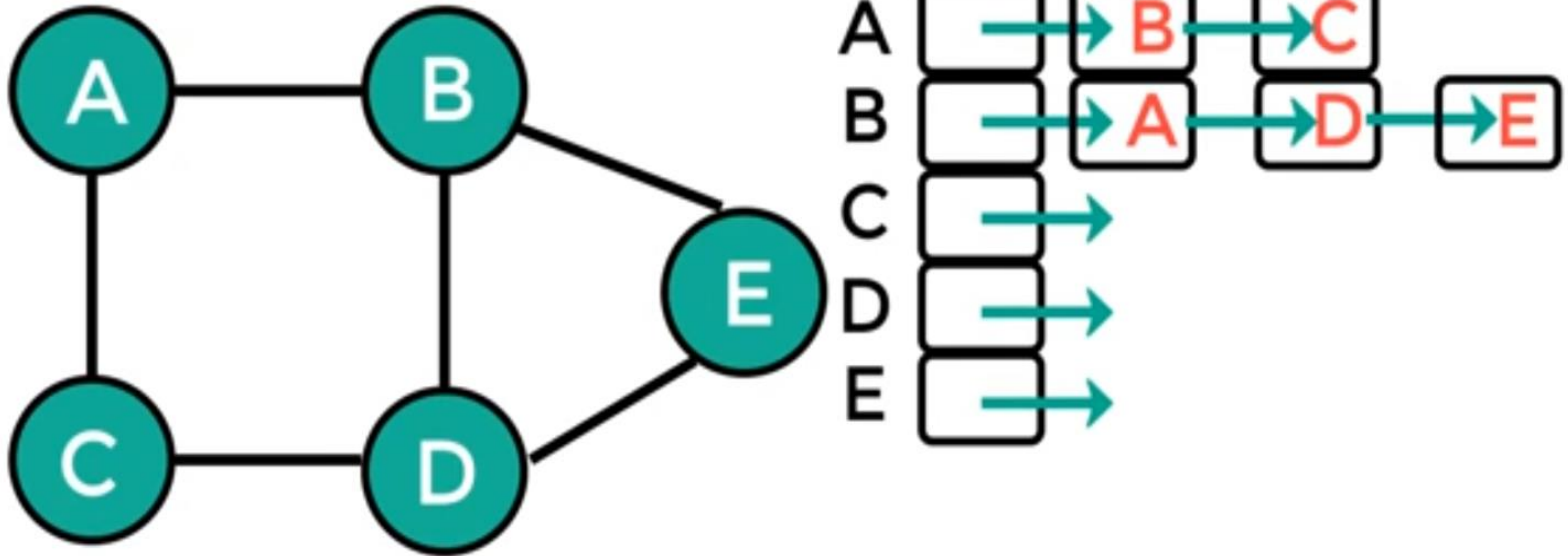


- في هذا التمثيل نستخدم **مصفوفة من القوائم المترابطة** **(array of linked lists)**:
  - حجم المصفوفة هو عدد الرؤوس في البيان n .
  - كل عنصر $A[i]$ في المصفوفة هو عبارة عن **قائمة مترابطة** تحوي جميع الرؤوس j التي ترتبط بالرأس i وفق الاتجاه من i إلى j
    - ✓ أي مجموعة **الرؤوس المجاورة** للرأس i.

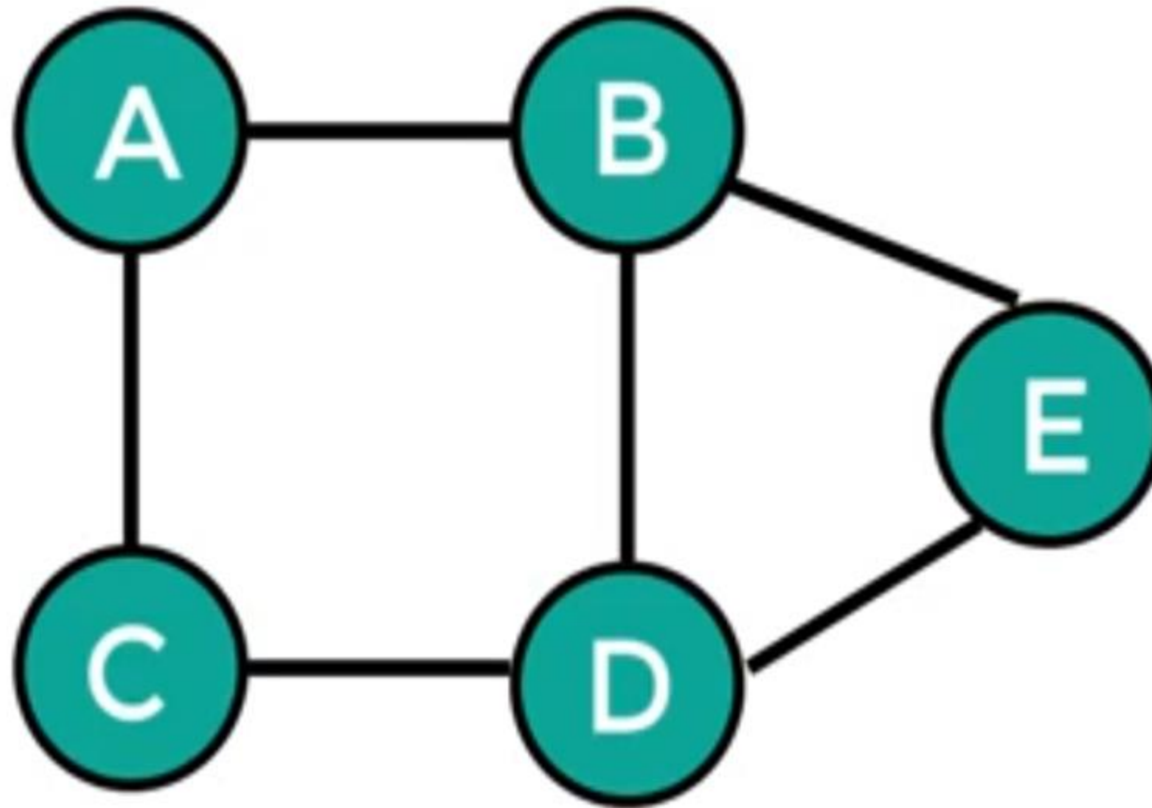
تمثيل البيان باستخدام قوائم التجاور

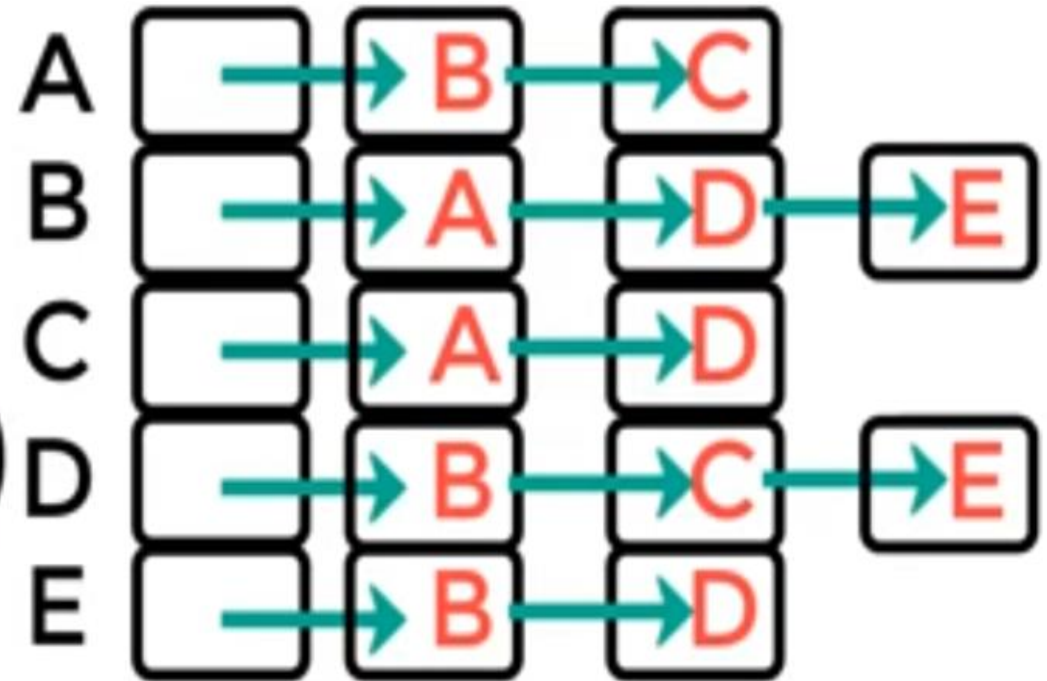


تمثيل البيان باستخدام قوائم التجاور



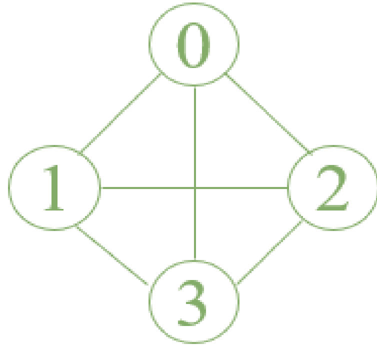
تمثيل البيان باستخدام قوائم التجاور

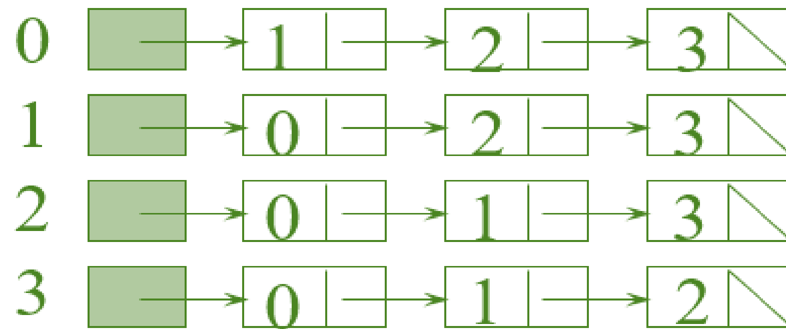




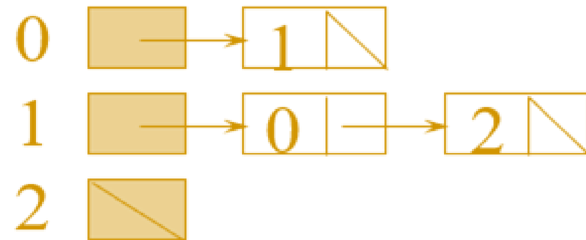
تمثيل البيان باستخدام قوائم التجاور

- أمثلة:



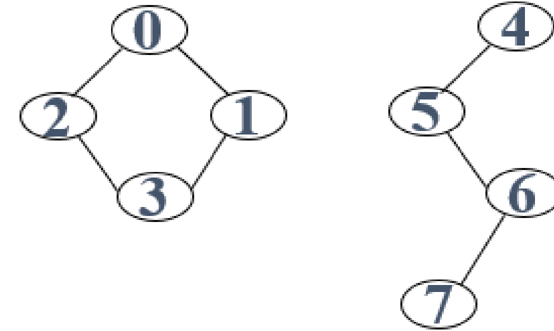


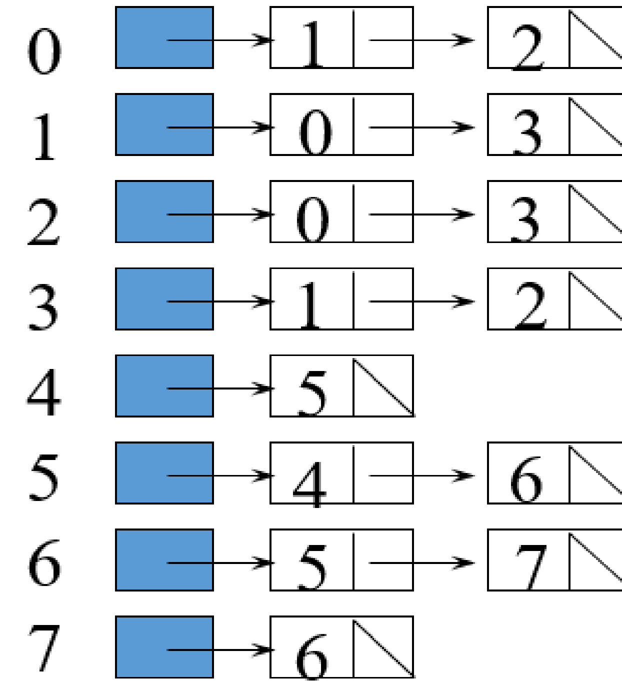
G_1





G_3

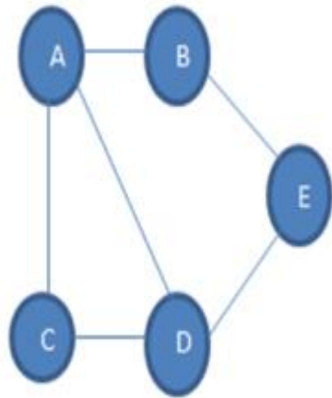


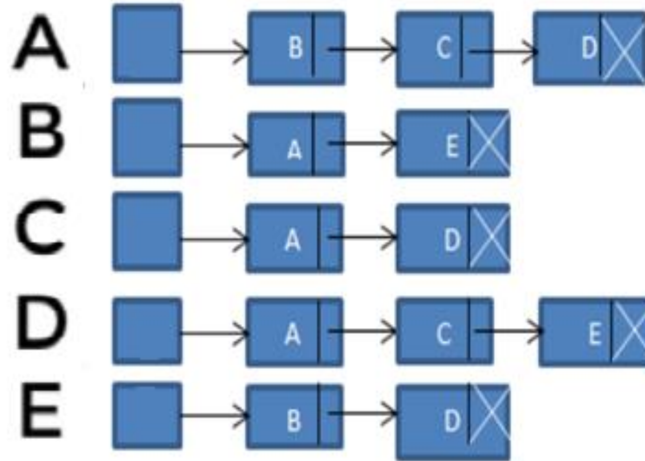


G_4

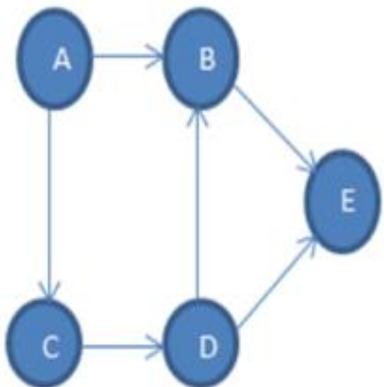
لحساب درجة رأس باستخدام قوائم التجاور

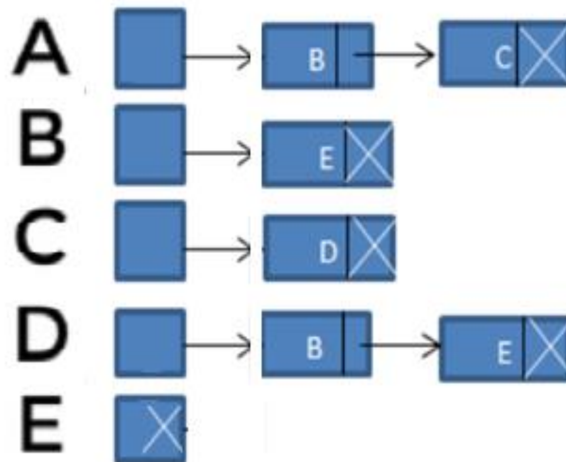
- تساوي عدد العقد في القائمة المترابطة الموافقة للرأس i
  - **درجة الرأس** i، في بيان **غير موجه** G ممثل بقوائم التجاور.





  - **الدرجة الصادرة** للرأس i، في بيان **موجه** G ممثل بقوائم التجاور.





مصفوفة التجاور – قوائم التجاور

	MATRIX	LIST
Space	$o(v^2)$	$o(v^2)$
Add Edge	o(1)	o(v)
Remove Edge	o(1)	o(v)
Query Edge	o(1)	o(v)
Find Neighbors	o(v)	o(v)
Add Node	$o(v^2)$	o(1)
Remove Node	$o(v^2)$	$o(v^2)$

Weighted Graph البيان الموزون

- في معظم التطبيقات التي تستخدم البيان لتمثيل بياناتها(data)، يخصص لأضلاع البيان أوزان .
  - قد تمثل هذه الأوزان **المسافة/** أو **كلفة الذهاب** من رأس إلى رأس آخر مجاور له .
- لتمثيل هذه الأوزان:
  - باستخدام مصفوفة التجاور، نستبدل القيمة 1 بوزن الضلع الموافق.
  - باستخدام قوائم التجاور، نضيف حقل الوزن (weight field) إلى بنية العقدة في القوائم المترابطة.

